Les « Listview » (approfondissement)

Introduction

Dans la section précédente, nous avons vu comment alimenter une ListView à l'aide d'un objet adaptateur. Android offre certains adaptateurs qu'on peut utiliser, mais qui ne permettent pas de créer toute sorte de ListView. Dans ce cours, nous allons apprendre à créer des adaptateurs personnalisés (custom adapters) qui offrent plus de liberté quant à alimenter et afficher la ListView.

Exemple 1 : adaptateur personnalisé qui hérite de la classe BaseAdapter

Dans cet exemple, nous afficherons une listView dont chaque ligne représente un produit. Une ligne est composée de l'image du produit, son nom et sa description. Pour ce faire, nous allons créer les fichiers suivants :

- Une classe produit avec trois attributs (nom, prix et description)
- Un fichier XML qui représente la forme de la ligne (ImageView et deux TextViews) : item_product_list.xml
- Une classe ProductListAdapter qui hérite de la classe BaseAdapter
- La classe MainActivity
- Le Layout activity_main.xml

Activity_main.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="64dp"
    android:paddingRight="64dp"
    android:paddingTop="16dp"
    tools:context="com.ehtp.rachik.customadptr.MainActivity">

    <ListView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentBottom="true"
        android:layout_alignParentEnd="true"
        android:layout_marginEnd="10dp"
        android:layout_marginBottom="10dp"
        android:id="@+id/listView" />
</RelativeLayout>
```

Item_productList.xml :

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:text="TextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/tv_name"
        android:textColor="@android:color/holo_blue_bright"
        tools:text="name" />

    <TextView
        android:text="TextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/tv_price"
        android:textColor="@android:color/holo_green_light"
        tools:text="price" />

    <TextView
        android:text="TextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/tv_description"
        android:textColor="@android:color/holo_red_light"
        tools:text="description" />
</LinearLayout>
```

Product.java :

```java
public class Product {

    private String name;
    private int price;
    private String description;

    public Product(String name, int price, String description) {
        this.name = name;
        this.price = price;
        this.description = description;
    }

    public String getDescription() { return description; }

    public int getPrice() { return price; }

    public String getName() { return name; }

}
```

ProductListAdapter.java :

```
public class ProductListAdapter extends BaseAdapter {

    private Context mContext;
    private List<Product> mListProduct;

    //Constructor

    public ProductListAdapter(Context mContext, List<Product> mListProduct) {
        this.mContext = mContext;
        this.mListProduct = mListProduct;
    }
    @Override
    public int getCount() { return mListProduct.size(); }
    @Override
    public Object getItem(int position) { return mListProduct.get(position); }
    @Override
    public long getItemId(int position) { return position; }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View v= View.inflate(mContext, R.layout.item_product_list,null);
        TextView name= (TextView) v.findViewById(R.id.tv_name);
        TextView price= (TextView) v.findViewById(R.id.tv_price);
        TextView description= (TextView) v.findViewById(R.id.tv_description);

        name.setText(mListProduct.get(position).getName());
        price.setText(String.valueOf(mListProduct.get(position).getPrice()));
        description.setText(mListProduct.get(position).getDescription());

        return v;
    }
}
```

MainActivity.java :

```java
ListView list;
ArrayList<Product> listPtoduct;
ProductListAdapter adapter;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    list= (ListView) findViewById(R.id.listView);
    listPtoduct= new ArrayList<>();
    listPtoduct.add(new Product("toshiba",350,"mljqsdmkj"));
    listPtoduct.add(new Product("boch",350,"mljqsdmkj"));
    listPtoduct.add(new Product("exp",350,"mljqsdmkj"));
    listPtoduct.add(new Product("exp2",350,"mljqsdmkj"));
    listPtoduct.add(new Product("sharp",350,"mljqsdmkj"));
    listPtoduct.add(new Product("HP",350,"mljqsdmkj"));
    listPtoduct.add(new Product("Nokia",350,"mljqsdmkj"));
    listPtoduct.add(new Product("LG",350,"mljqsdmkj"));
    listPtoduct.add(new Product("toshiba",350,"mljqsdmkj"));
    listPtoduct.add(new Product("boch",350,"mljqsdmkj"));
    listPtoduct.add(new Product("exp2",350,"mljqsdmkj"));
    listPtoduct.add(new Product("exp",350,"mljqsdmkj"));
    listPtoduct.add(new Product("sharp",350,"mljqsdmkj"));
    listPtoduct.add(new Product("HP",350,"mljqsdmkj"));
    listPtoduct.add(new Product("Nokia",350,"mljqsdmkj"));
    listPtoduct.add(new Product("LG",350,"mljqsdmkj"));
    adapter= new ProductListAdapter(getApplicationContext(),listPtoduct);
    list.setAdapter(adapter);
```

Exemple 2 : adaptateur personnalisé qui hérite de la classe ArrayAdapter

Activity_main.xml :

Dans cet exemple, nous afficherons une ListView dont chaque ligne représente un animal. Une ligne est composée de l'image du produit, son nom et sa description.

Pour ce faire nous allons créer les fichiers suivants :

- Une classe Animal qui possède trois attributs (image, nom et description)
- Un Layout XML qui représente une ligne de la ListView : list_item.xml
- Une classe CustomAdapter qui hérite de ArrayAdapter
- MainActivity.java
- activity_main.xml

Les images que nous utiliserons seront stockées dans le fichier Drawable.

Ci-dessous une vue de la structure de notre projet :

- app
  - manifests
  - java
    - com.ehtp.rachik.customadapter2
      - Animal
      - CustomAdapter
      - MainActivity
    - com.ehtp.rachik.customadapter2 (androidTest)
    - com.ehtp.rachik.customadapter2 (test)
  - res
    - drawable
      - a.jpg
      - b.jpg
      - c.jpg
      - e.jpeg
      - f.jpg
    - layout
      - activity_main.xml
      - list_item.xml
    - mipmap
    - values

MainActivity.java :

```java
public class MainActivity extends AppCompatActivity {
    ArrayList<Animal> AnimalList;
    ListView maListe;
    CustomAdapter adapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        maListe=(ListView) findViewById(R.id.listview);

        AnimalList= new ArrayList<Animal>();
        AnimalList.add(new Animal(R.drawable.a,"exemple de nom1","exemple de description1"));
        AnimalList.add(new Animal(R.drawable.b,"exemple de nom2","exemple de description2"));
        AnimalList.add(new Animal(R.drawable.c,"exemple de nom3","exemple de description3"));
        AnimalList.add(new Animal(R.drawable.e,"exemple de nom4","exemple de description4"));
        AnimalList.add(new Animal(R.drawable.f,"exemple de nom5","exemple de description5"));
        AnimalList.add(new Animal(R.drawable.a,"exemple de nom1","exemple de description1"));
        AnimalList.add(new Animal(R.drawable.b,"exemple de nom2","exemple de description2"));
        AnimalList.add(new Animal(R.drawable.c,"exemple de nom3","exemple de description3"));
        AnimalList.add(new Animal(R.drawable.e,"exemple de nom4","exemple de description4"));
        AnimalList.add(new Animal(R.drawable.f,"exemple de nom5","exemple de description5"));
        AnimalList.add(new Animal(R.drawable.a,"exemple de nom1","exemple de description1"));
        AnimalList.add(new Animal(R.drawable.b,"exemple de nom2","exemple de description2"));
        AnimalList.add(new Animal(R.drawable.c,"exemple de nom3","exemple de description3"));
        AnimalList.add(new Animal(R.drawable.e,"exemple de nom4","exemple de description4"));
        AnimalList.add(new Animal(R.drawable.f,"exemple de nom5","exemple de description5"));

        adapter= new CustomAdapter(getApplicationContext(), AnimalList);
        maListe.setAdapter(adapter);
```

Activity_main.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="16dp"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    tools:context="com.ehtp.rachik.customadapter2.MainActivity">

    <ListView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentTop="true"
        android:layout_marginTop="24dp"
        android:id="@+id/listview"
        android:layout_alignParentEnd="true" />
</RelativeLayout>
```

Animal.java :

```
public class Animal {
    private int mImage;
    private String mName;
    private String mDescription;

    public Animal(int mImage, String mName, String mDescription) {
        this.mImage = mImage;
        this.mName = mName;
        this.mDescription = mDescription;
    }

    public int getmImage() { return mImage; }

    public String getmName() { return mName; }

    public String getmDescription() { return mDescription; }
}
```

list_item.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal" android:layout_width="match_parent"
    android:layout_height="match_parent">

    <ImageView
        android:id="@+id/image"
        android:layout_gravity="left"
        android:layout_width="50dp"
        android:layout_height="50dp"
        android:layout_marginTop="5dp"
         />
    <LinearLayout
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:gravity="right"
        android:orientation="vertical">
    <TextView
        android:text="TextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="left"
        android:textSize="10sp"
        android:textStyle="bold"
        android:textColor="@android:color/black"
        android:id="@+id/textView"
         />
    <TextView
        android:text="TextView"
```

CustomAdapter.java :

```java
public class CustomAdapter extends ArrayAdapter<Animal> {
    ArrayList<Animal> mList;
    Context mContext;
    public CustomAdapter(Context context, ArrayList<Animal> mList) {
        super(context,0,mList);
        this.mList=mList;
        this.mContext=context;
    }
    @NonNull
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View v= convertView;
        if(v==null) {
            v = LayoutInflater.from(mContext).inflate(R.layout.list_item, parent, false);
        }

            Animal animalCourant = mList.get(position);
            ImageView image= (ImageView)v.findViewById(R.id.image);
            image.setImageResource(animalCourant.getmImage());
            TextView name=(TextView)v.findViewById(R.id.textView);
            name.setText(animalCourant.getmName());
            TextView description=(TextView)v.findViewById(R.id.textView2);
            description.setText(animalCourant.getmDescription());
        return v;

    }
}
```

ViewHolder pattern

Poussé par Google, le patron de conception ViewHolder est devenu l’architecture utilisée pour obtenir un défilement rapide et fluide.

L’objectif est de proposer une façon plus élégante et plus pérenne pour construire des listes de vues efficaces en séparant les responsabilités.

Implémentation de la classe CustomView Avec ViewHolder :

```
public class CustomAdapter extends ArrayAdapter<Animal> {
    ArrayList<Animal> mList;
    Context mContext;
    public CustomAdapter(Context context, ArrayList<Animal> mList) {
        super(context,0,mList);
        this.mList=mList;
        this.mContext=context; }
    static class ViewHolder{
         ImageView mImage;
         TextView mName;
         TextView mDescription;}
    @NonNull
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        View v= convertView;
        ViewHolder vHolder;
        if(v==null) { v = LayoutInflater.from(mContext).inflate(R.layout.list_item, parent, false);
                      vHolder = new ViewHolder();
                      vHolder.mDescription= (TextView) v.findViewById(R.id.textView2);
                      vHolder.mName=(TextView) v.findViewById(R.id.textView);
                      vHolder.mImage=(ImageView) v.findViewById(R.id.image);
                      v.setTag(vHolder);                    }
        else        { vHolder=(ViewHolder)v.getTag();    }
            Animal animalCourant = mList.get(position);
            ImageView image= vHolder.mImage;
            image.setImageResource(animalCourant.getmImage());
            TextView name=vHolder.mName;
            name.setText(animalCourant.getmName());
            TextView description=vHolder.mDescription;
            description.setText(animalCourant.getmDescription());
        return v;
    }
```